**[IfcAlignment 0.0.1]**

**Alignment Description**

**EXPRESS Format Specification**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Function | Date | Signature |
| Written by | Julian Amann | Team member | 07/11/2013 |  |
|  |  |  |  |  |
|  |  |  |  |  |

Changes

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Data | Writer(s) | Adds to the previous version |
| 0.0.1 | 07/11/2013 | Julian Amann | First draft. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Summary

[1 Introduction 4](#_Toc375597086)

[2 General Description 5](#_Toc375597087)

[2.1 Content 5](#_Toc375597088)

[2.2 Requirements for aN alignment data structure 5](#_Toc375597089)

[2.3 A proposal for a new alignment data structure 5](#_Toc375597090)

[2.4 Computation of vertical alignments 8](#_Toc375597091)

[3 Detailed description 10](#_Toc375597092)

[3.1 IfcReferenceCurve 10](#_Toc375597093)

[3.2 IfcReferenceCurve3D 10](#_Toc375597094)

[3.3 IfcReferenceCurveAlignment2D 11](#_Toc375597095)

[3.4 IfcHorizontalAlignment 11](#_Toc375597096)

[3.5 IfcHorizontalAlignmentSegment 12](#_Toc375597097)

[3.6 IfcHorizontalAlignmentLine 12](#_Toc375597098)

[3.7 IfcHorizontalAlignmentCircularSegment 13](#_Toc375597099)

[3.8 IfcClothoidTypeEnum 14](#_Toc375597100)

[3.9 IfcClothoid 14](#_Toc375597101)

[3.10 IfcHorizontalAlignmentClothoid 15](#_Toc375597102)

[3.11 IfcTransitionCurve 15](#_Toc375597103)

[3.12 IfcVerticalAlignment 15](#_Toc375597104)

[3.13 IfcVerticalAlignmentSegment 16](#_Toc375597105)

[3.14 IfcVerticalAlignmentPointVerticalInterscection 17](#_Toc375597106)

[3.15 IfcVerticalAlignmentRounding 17](#_Toc375597107)

[3.16 IfcVerticalAlignmentParabola 17](#_Toc375597108)

[4 Apendix 19](#_Toc375597109)

[4.1 EXPRESS Schema 19](#_Toc375597110)

[4.2 Example Step File 20](#_Toc375597111)

# Introduction

There is a great need for data exchange in the architecture, engineering and construction (AEC) industry. A study from NIST in 2004 showed that more than 15.8 billion dollars are wasted each year due to insufficient data exchange in building projects (Venugopal et al. 2012). A proportionally similar amount of money wastage is likely in the field of infrastructure projects.

At present, there is very limited data exchange support for infrastructure elements such as roads, bridges or tunnels. For building construction, however, the Industry Foundation Classes (IFC) (buildingSMART 2013) have already become established as a mature standard. Something similar is currently lacking in the infrastructure sector. In this document we describe, an alignment model that supports the well-established approach of aligning design based on vertical and horizontal alignments. The model supports also 3D space curves for alignments, which can be used, for instance, for alignment recording by GPS. We show how the IFC 4 standard can be extended with our alignment model. Furthermore, we describe how the alignment data model can be interconnected with IfcBridge.

# General Description

## Content

IFC-Bridge is an extension of the IFC Standard which is currently in ongoing development (Yakubi et al. 2006; Lebegue et al. 2012). It extends the current IFC data product model to include items of bridge design. A crucial part of the bridge design is the alignment curve. For this, the authors introduce an IfcReferenceCurveAlignment2D element which references a horizontal and a vertical alignment curve. For both the horizontal and vertical alignment, an IfcCurve element is used. Since IfcCurve is very general, it allows many different types of descriptions of curves, which makes it hard for software application implementers to handle all types and combinations of curve types. For example, editing the start or end radius of a clothoid is also difficult if it is described by arbitrary curve elements.

This document describes a generalized IFC 4 based alignment model that can be used in the field of infrastructure to describe road, tunnel and bridge alignments. The demonstrated model can be easily extended and used in other infrastructure domains such as bridge or road design. A proposal of an extended IFC 4 standard is also given and it is shown how it can be mapped to existing IFC 4 elements.

## Requirements for aN alignment data structure

An alignment data structure has to fulfill several requirements to be useful for infrastructure modeling. Since the user should be provided with the possibility to apply the well-established approach of using horizontal and vertical alignments besides directly storing a 3D curve, the corresponding vertical and horizontal alignment elements need to be stored.

The data structure must be able to support the drawing and modification of horizontal and vertical alignments. Moreover, it needs to be possible to generate a 3D space curve based on these alignment representations. Furthermore, it needs to be simple for software developers to adapt and integrate the alignment product data model to their software products. Since the presented alignment data model is based on the IFC 4 standard, it should re-use as many data structures from the existing standard as possible and should not duplicate data structures already present. The alignment data product model should also contain only necessary data and not data that can be derived. Additional data such as stationing also need to be included in the data model.

## A proposal for a new alignment data structure

First we consider an alignment in its top-level view. An alignment can be a 3D reference curve or a 2D alignment consisting of a horizontal and a vertical alignment. For the latter, some implementation defined restrictions have to be characterized. A horizontal alignment does not contain any junctions or gaps. Junctions are forbidden as a connection with a vertical alignment would thus be impossible. In the case of a junction, it is not clear how a proper vertical alignment can be developed. Gaps in a horizontal alignment are also not allowed because in this case we would also have to store this gap in the vertical alignment which would only make the data model more complicated. For junctions and gaps, another alignment with its own horizontal and vertical alignment has to be created. So instead of creating an alignment with a horizontal and a vertical alignment with a gap, we create two alignments each with its own junction and gap free horizontal and vertical alignment. This restriction simplifies the presented data model and does nevertheless support gaps and junctions. Since the introduced horizontal alignment and also the vertical alignment are gap free, the end point of an alignment segment is always the start point of a following alignment segment. So the same point is referenced two times, one time as start- and another time as endpoint. But this is acceptable considering the memory consumption (references are cheap). Moreover, storing just one reference would make the model cumbersome to use.

From the horizontal and vertical alignment, a 3D space curve can be computed. In the simple case where the horizontal and vertical alignments have the same length, the corresponding chainage in the horizontal and vertical alignment just have to be found. If the vertical alignment has a different length, it is not invalid. A proportionality factor will be computed and the vertical alignment will be sized so it has the same length as the horizontal alignment. The horizontal alignment stores also the start chainage. The following chainages can be computed by summing up the different lengths of the horizontal alignment segments.

Figure 5 shows an overview of the proposed alignment data structure. On the top level there is the *IfcReferenceCurve* element. The model supports a 3D space curve (*IfcReferenceCurve3D*) as well as the traditional approach of horizontal and vertical alignments (*IfcReferenceAlignment2D*). The IfcReferenceAlignment2D consist of a gap and junction free horizontal (*IfcHorizontalAlignment*) and vertical alignment (*IfcVerticalAlignment*). The *IfcHorizontalAlignment* consist of an order list of *IfcHorizontal- AlingmentSegments*. An *IfcHorizontalAlingmentSegment* is a superclass of *IfcHorizontalAlignmentLine* for line segments, *IfcHorizontalAlignmentCircularSegment* for circle segments and *IfcHorizontalAlignmentTransiton- Curve* for transition curves. The only supported transition curve is the *IfcHorizontalAlignmentClothoid* for a clothoid.

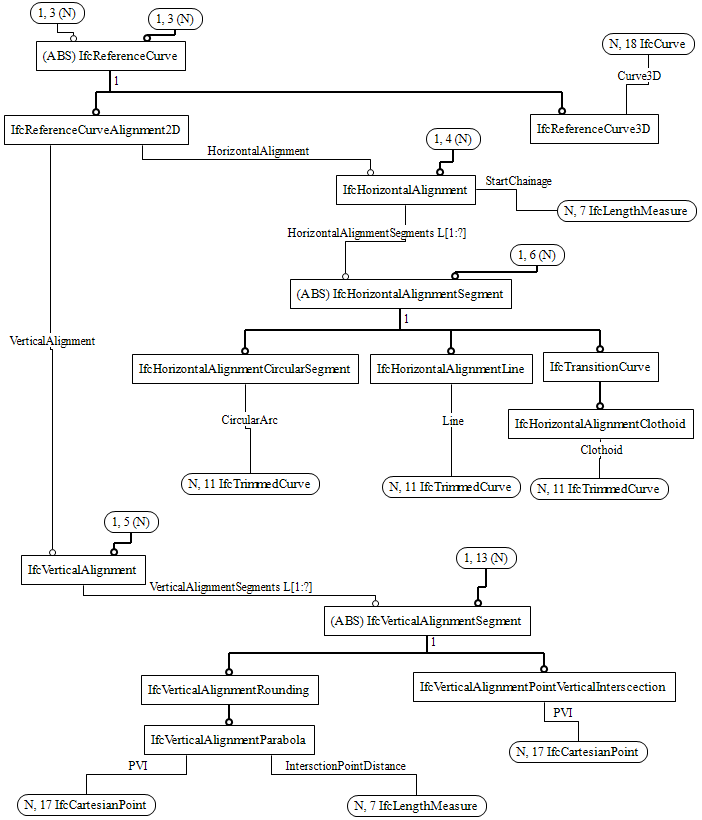


Figure 5. EXPRESS G-Diagram giving an overview of the proposed alignment data structure.

The vertical alignment is stored in the *IfcVerticalAlignment* element. It consists of an ordered list *IfcVerticalAlignmentSegments* such as *IfcVerticalAlignmentPointVerticalIntersection* and *IfcVertical- AlignmentRounding*. An *IfcVerticalAlignmentRounding* has only one subclass (*IfcVerticalAlignmentParabola*).

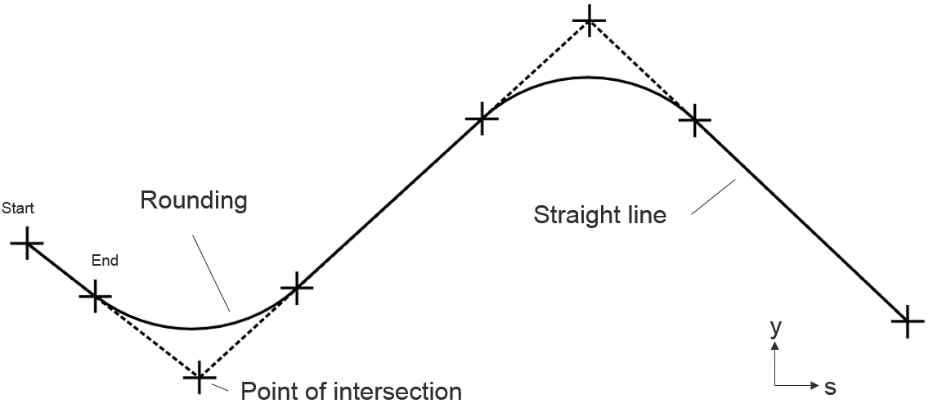
A horizontal alignment line segment can be described by a trimmed curve (*IfcTrimmedCurve*) that is using an *IfcLine* as a basis curve. The start point is used as the origin of the *IfcLine* and the difference vector between the end- and start point is used as direction vector. The first and second trim points can be set to the start and end of the corresponding line. The horizontal alignment circular segment also uses a trimmed curve. *IfcCircle* is used as a basis curve for the trimmed curve. The first and the second trim point are again set to the start and end point of the arc. The *IfcTrimmedCurve* offers a sense agreement which can be used to store if the circle is clockwise or counterclockwise. A horizontal alignment clothoid curve segment also has a start and an end point. These points are stored again in the first and the second trim point of the *IfcTrimmedCurve*. The sense argument of the trimmed curve is again used to store the rotation sense of the clothoid. Currently, there is no *IfcClothoid* curve in the IFC standard which can be used as a basis curve. An *IfcClothoid* element can easily be introduced in the IFC standard to circumvent this problem. Table 1 shows in detail the mapping of the horizontal alignment line and circle segment to an ASCII STEP file.

Table 1. Mapping of alignment segments to IFC4/STEP

|  |  |
| --- | --- |
| IfcHorizontalAlignmentLine | IfcHorizontalAlignmentCircularSegment |
| #2=IFCHORIZONTALALIGNMENTLINE(#3);  #3=IFCTRIMMEDCURVE(#4,(#8),(#9),,$);  #4=IFCLINE(#5,#6);  #5=IFCCARTESIANPOINT((1031.95,1177.96));  #6=IFCVECTOR(#7,1);  #7=IFCDIRECTION((76.1796,252.095));  #8=IFCCARTESIANPOINT((1031.95,1177.96));  #9=IFCCARTESIANPOINT((1108.13,1430.06)); | #11=IFCHORIZONTALALIGNMENT  CIRCULARSEGMENT(#12);  #12=IFCTRIMMEDCURVE(#13,(#16),(#17),.T.,$);  #13=IFCCIRCLE(#14,86.6106);  #14=IFCAXIS2PLACEMENT2D(#15,$);  #15=IFCCARTESIANPOINT((1254.2,1394.62));  #16=IFCCARTESIANPOINT((1113.24,1445.92));  #17=IFCCCARTESIANPOINT((1165.1,1515.29)); |

## Computation of vertical alignments

A vertical alignment consist of rounding an straight line segments.



The start point of a vertical alignment is a so called point of vertical intersection. This name is chosen because connected points of vertical intersection describe gradient lines. Typically, a construction engineer drafts these lines first and then replaces the discontinuous transitions with roundings, which are typically parabolas. From the intersection points and the distance of the left PVC and right PVT point (see Figure 4) the PVT and PVC can be calculated. First the slope *m* of the first line segment (PointVerticalIntersction.point to Parabola.pointVerticalIntersection) is computed. Given the PVI point of the parabola and the slope *m*, the y-intercept can be calculated. The length between the PVT and PVC is called the parabola span. The x-value of PVT can be computed by Equation (1). The corresponding y-coordinate can then also be calculated.

(1)

After computing these points, the vertical alignment can be viewed in terms of line segments and parabolas instead of points of vertical intersections and roundings. While the latter view is effectively more useful when a 3D space curve needs to be computed from the horizontal and vertical alignment, the intersection points are very important for editing a vertical alignment. For this reason, this approach has been chosen to store the vertical alignment data.

# Detailed description

This chapter describes all the EXPRESS elements of the IfcAlignment file in logical top down order.

## IfcReferenceCurve

### Concepts

The <IfcReferenceCurve> is can be either be a 2D based alignment model (with a horizontal and a vertical alignment) or a 3d space curve.

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcReferenceElement> |
| **Child elements** |  |
| Subtypes | <IfcReferenceCurve3D, IfcReferenceCurveAlignment2D> |

### EXPRESS specification

ENTITY IfcReferenceCurve

ABSTRACT SUPERTYPE OF (ONEOF

(IfcReferenceCurve3D,IfcReferenceCurveAlignment2D))

SUBTYPE OF (IfcReferenceElement);

END\_ENTITY;

### Example

#0=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#1,#23);

## IfcReferenceCurve3D

### Concepts

The <IfcReferenceCurve3D> describes a 3d space curve.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| Curve3D | IfcCurve | A 3d space curve. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcReferenceElement> |
| **Child elements** | <IfcCurve> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcReferenceCurve3D

SUBTYPE OF (IfcReferenceCurve);

Curve3D : IfcCurve;

END\_ENTITY;

## IfcReferenceCurveAlignment2D

### CONCEPTS

The traditional approach of horizontal and vertical alignments (IfcReferenceAlignment2D). The IfcReferenceAlignment2D consist of a gap and junction free horizontal (IfcHorizontalAlignment) and vertical alignment (IfcVerticalAlignment).

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| HorizontalAlignment | IfcHorizontalAlignment | The horizontal alignment. |
| VerticalAlignment | IfcVerticalAlignment | The vertical alignment. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcReferenceElement> |
| **Child elements** | <IfcHorizontalAlignment>  <IfcVerticalAlignment> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcReferenceCurveAlignment2D

SUBTYPE OF (IfcReferenceCurve);

HorizontalAlignment : IfcHorizontalAlignment;

VerticalAlignment : IfcVerticalAlignment;

END\_ENTITY;

### Example

#34=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#35,#97);

## IfcHorizontalAlignment

### CONCEPTS

The *IfcHorizontalAlignment* consist of an order list of *IfcHorizontal- AlingmentSegments*.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| HorizontalAlignmentSegments | List<IfcHorizontalAlignmentSegment> | Horizontal alignment segments. |
| StartChainage | IfcLengthMeasure | Start station. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcReferenceElement> why? |
| **Child elements** | <IfcHorizontalAlignmentSegment>  <IfcLengthMeasure> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcHorizontalAlignment

SUBTYPE OF (IfcReferenceElement);

HorizontalAlignmentSegments : LIST [1:?] OF IfcHorizontalAlignmentSegment;

StartChainage : IfcLengthMeasure;

END\_ENTITY;

## IfcHorizontalAlignmentSegment

### CONCEPTS

An *IfcHorizontalAlingmentSegment* is a superclass of *IfcHorizontalAlignmentLine* for line segments, *IfcHorizontalAlignmentCircularSegment* for circle segments and *IfcHorizontalAlignmentTransiton- Curve* for transition curves.

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcCurve> |
| **Child elements** |  |
| Subtypes | < IfcHorizontalAlignmentLine>  < IfcHorizontalAlignmentCircularSegment>  < IfcTransitionCurve> |

### EXPRESS specification

ENTITY IfcHorizontalAlignmentSegment

ABSTRACT SUPERTYPE OF (ONEOF

(IfcHorizontalAlignmentLine,IfcHorizontalAlignmentCircularSegment,IfcTransitionCurve))

SUBTYPE OF (IfcCurve);

END\_ENTITY;

## IfcHorizontalAlignmentLine

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| Line | IfcTrimmedCurve | A straight line. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcHorizontalAlignmentSegment> |
| **Child elements** | <IfcTrimmedCurve> |
| Subtypes |  |

### CONCEPTS

A line segment.

### EXPRESS specification

ENTITY IfcHorizontalAlignmentLine

SUBTYPE OF (IfcHorizontalAlignmentSegment);

Line : IfcTrimmedCurve;

END\_ENTITY;

## IfcHorizontalAlignmentCircularSegment

### CONCEPTS

A circular arc segment.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| CircularArc | IfcTrimmedCurve | An arc. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcHorizontalAlignmentSegment> |
| **Child elements** | <IfcTrimmedCurve> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcHorizontalAlignmentCircularSegment

SUBTYPE OF (IfcHorizontalAlignmentSegment);

CircularArc : IfcTrimmedCurve;

END\_ENTITY;

## IfcClothoidTypeEnum

### CONCEPTS

Used to describe a clothoid with an infinite or finite start/end radius.

### EXPRESS specification

TYPE IfcClothoidTypeEnum = ENUMERATION OF

(START\_RADIUS\_INFINITE

,END\_RADIUS\_INFINITE

,START\_AND\_END\_FINITE);

END\_TYPE;

## IfcClothoid

### CONCEPTS

A clothoid. TODO a clothoid can be described by six parameters.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| PI | IfcTrimmedCurve | Point of intersection. |
| ClothoidConstant | IfcLengthMeasure | Constant of clothoid. |
| RadiusStart | IfcLengthMeasure | Start radius of clothoid. |
| RadiusEnd | IfcLengthMeasure | End radius of clothoid. |
| Length | IfcLengthMeasure | Length of clothoid. |
| Type | IfcClothoidTypeEnum | Type of clothoid. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcHorizontalAlignmentSegment> |
| **Child elements** | <IfcCartesianPoint>  <IfcLengthMeasure>  <IfcClothoidTypeEnum>  <IfcTrimmedCurve> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcClothoid

SUBTYPE OF (IfcCurve);

PI : IfcCartesianPoint;

ClothoidConstant : IfcLengthMeasure;

RadiusStart : IfcLengthMeasure;

RadiusEnd : IfcLengthMeasure;

Length : IfcLengthMeasure;

Type : IfcClothoidTypeEnum;

END\_ENTITY;

## IfcHorizontalAlignmentClothoid

### CONCEPTS

A clothoid used in alignment.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| Clothoid | IfcTrimmedCurve | Geometric representation of a clothoid. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | < IfcTransitionCurve> |
| **Child elements** |  |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcHorizontalAlignmentClothoid

SUBTYPE OF (IfcTransitionCurve);

Clothoid : IfcTrimmedCurve;

END\_ENTITY;

## IfcTransitionCurve

### CONCEPTS

A transition curve. This proposal defines only a clothoid as a transition curve, but there are many other possibilities.

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcHorizontalAlignmentSegment> |
| **Child elements** | <IfcHorizontalAlignmentClothoid> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcTransitionCurve

SUBTYPE OF (IfcHorizontalAlignmentSegment);

END\_ENTITY;

## IfcVerticalAlignment

### CONCEPTS

The vertical alignment is stored in the *IfcVerticalAlignment* element. It consists of an ordered list *IfcVerticalAlignmentSegments* such as *IfcVerticalAlignmentPointVerticalIntersection* and *IfcVertical- AlignmentRounding*.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| VerticalAlignmentSegments | List<IfcVerticalAlignmentSegment> | List of vertical alignments. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | < IfcReferenceElement> why? |
| **Child elements** | <IfcVerticalAlignmentSegment> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcVerticalAlignment

SUBTYPE OF (IfcReferenceElement);

VerticalAlignmentSegments : LIST [1:?] OF IfcVerticalAlignmentSegment;

END\_ENTITY;

## IfcVerticalAlignmentSegment

### CONCEPTS

*A IfcVerticalAlignmentSegments is either a* *IfcVerticalAlignmentPointVerticalIntersection* or *IfcVertical- AlignmentRounding*.

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcCurve> why? |
| **Child elements** | <IfcVerticalAlignmentSegment> |
| Subtypes | <IfcVerticalAlignmentPointVerticalInterscection>  <IfcVerticalAlignmentRounding> |

### EXPRESS specification

ENTITY IfcVerticalAlignmentSegment

ABSTRACT SUPERTYPE OF (ONEOF

(IfcVerticalAlignmentPointVerticalInterscection,IfcVerticalAlignmentRounding))

SUBTYPE OF (IfcCurve);

END\_ENTITY;

## IfcVerticalAlignmentPointVerticalInterscection

### CONCEPTS

A point of vertical intersection.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| PVI | IfcCartesianPoint | Point of vertical intersection. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcVerticalAlignmentSegment> |
| **Child elements** | <IfcCartesianPoint> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcVerticalAlignmentPointVerticalInterscection

SUBTYPE OF (IfcVerticalAlignmentSegment);

PVI : IfcCartesianPoint;

END\_ENTITY;

## IfcVerticalAlignmentRounding

### CONCEPTS

A vertical alignment rounding element.

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcVerticalAlignmentSegment> |
| **Child elements** | <IfcCartesianPoint> |

### EXPRESS specification

ENTITY IfcVerticalAlignmentRounding

ABSTRACT SUPERTYPE OF (ONEOF

(IfcVerticalAlignmentParabola))

SUBTYPE OF (IfcVerticalAlignmentSegment);

END\_ENTITY;

## IfcVerticalAlignmentParabola

### Concepts

A parabola element in a vertical alignment.

### ATTRIBUTES

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| PVI | IfcCartesianPoint | Point of vertical intersection. |
| IntersectionPointDistance | IfcLengthMeasure | Span of parabola. |

### Related Elements

|  |  |
| --- | --- |
| Parent elements | <IfcVerticalAlignmentRounding> |
| **Child elements** | <IfcCartesianPoint>  <IfcLengthMeasure> |
| Subtypes |  |

### EXPRESS specification

ENTITY IfcVerticalAlignmentParabola

SUBTYPE OF (IfcVerticalAlignmentRounding);

PVI : IfcCartesianPoint;

IntersectionPointDistance : IfcLengthMeasure;

END\_ENTITY;

# Apendix

## EXPRESS Schema

ENTITY IfcAlignmentElement

ABSTRACT SUPERTYPE OF (ONEOF

(IfcReferenceElement))

SUBTYPE OF (IfcProduct);

END\_ENTITY;

ENTITY IfcReferenceElement

ABSTRACT SUPERTYPE OF (ONEOF

(IfcReferenceCurve))

SUBTYPE OF (IfcAlignmentElement);

END\_ENTITY;

ENTITY IfcReferenceCurve

ABSTRACT SUPERTYPE OF (ONEOF

(IfcReferenceCurve3D,IfcReferenceCurveAlignment2D))

SUBTYPE OF (IfcReferenceElement);

END\_ENTITY;

ENTITY IfcReferenceCurve3D

SUBTYPE OF (IfcReferenceCurve);

Curve3D : IfcCurve;

END\_ENTITY;

ENTITY IfcReferenceCurveAlignment2D

SUBTYPE OF (IfcReferenceCurve);

HorizontalAlignment : IfcHorizontalAlignment;

VerticalAlignment : IfcVerticalAlignment;

END\_ENTITY;

ENTITY IfcHorizontalAlignment

SUBTYPE OF (IfcReferenceElement);

HorizontalAlignmentSegments : LIST [1:?] OF IfcHorizontalAlignmentSegment;

StartChainage : IfcLengthMeasure;

END\_ENTITY;

ENTITY IfcHorizontalAlignmentSegment

ABSTRACT SUPERTYPE OF (ONEOF

(IfcHorizontalAlignmentLine,IfcHorizontalAlignmentCircularSegment,IfcTransitionCurve))

SUBTYPE OF (IfcCurve);

END\_ENTITY;

ENTITY IfcHorizontalAlignmentLine

SUBTYPE OF (IfcHorizontalAlignmentSegment);

Line : IfcTrimmedCurve;

muh : IfcCartesianPoint;

END\_ENTITY;

ENTITY IfcHorizontalAlignmentCircularSegment

SUBTYPE OF (IfcHorizontalAlignmentSegment);

CircularArc : IfcTrimmedCurve;

END\_ENTITY;

TYPE IfcClothoidTypeEnum = ENUMERATION OF

(START\_RADIUS\_INFINITE

,END\_RADIUS\_INFINITE

,START\_AND\_END\_FINITE);

END\_TYPE;

ENTITY IfcClothoid

SUBTYPE OF (IfcCurve);

PI : IfcCartesianPoint;

ClothoidConstant : IfcLengthMeasure;

RadiusStart : IfcLengthMeasure;

RadiusEnd : IfcLengthMeasure;

Length : IfcLengthMeasure;

Type : IfcClothoidTypeEnum;

END\_ENTITY;

ENTITY IfcHorizontalAlignmentClothoid

SUBTYPE OF (IfcTransitionCurve);

Clothoid : IfcTrimmedCurve;

END\_ENTITY;

ENTITY IfcTransitionCurve

SUBTYPE OF (IfcHorizontalAlignmentSegment);

END\_ENTITY;

## Example Step File

In the following code listing an STEP file is shown that is using the previously shown Alignment EXPERSS Schema.

ISO-10303-21;

HEADER;

FILE\_DESCRIPTION(('IFC4'),'2;1');

FILE\_NAME('IfcGears-export.ifc','2013-11-21T09:18:37',(''),('',''),'','IfcGears','');

FILE\_SCHEMA(('IFC4'));

ENDSEC;

DATA;

#0=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#1,#23);

#1=IFCHORIZONTALALIGNMENT($,$,$,$,$,$,$,(#2,#9,#16),0);

#2=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#3);

#3=IFCTRIMMEDCURVE(#4,(#7),(#8),.F.,$);

#4=IFCCIRCLE(#5,15.00004036907433);

#5=IFCAXIS2PLACEMENT2D(#6,$);

#6=IFCCARTESIANPOINT((3513050.8722999999,5516206.7060000002));

#7=IFCCARTESIANPOINT((3513037.9591000001,5516199.0738000004));

#8=IFCCARTESIANPOINT((3513063.4561999999,5516198.5422999999));

#9=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#10);

#10=IFCTRIMMEDCURVE(#11,(#14),(#15),.F.,$);

#11=IFCCIRCLE(#12,15.000017896838163);

#12=IFCAXIS2PLACEMENT2D(#13,$);

#13=IFCCARTESIANPOINT((3513050.8722999999,5516206.7060000002));

#14=IFCCARTESIANPOINT((3513063.4561999999,5516198.5422999999));

#15=IFCCARTESIANPOINT((3513051.6502999999,5516221.6858000001));

#16=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#17);

#17=IFCTRIMMEDCURVE(#18,(#21),(#22),.F.,$);

#18=IFCCIRCLE(#19,14.99998973451533);

#19=IFCAXIS2PLACEMENT2D(#20,$);

#20=IFCCARTESIANPOINT((3513050.8722999999,5516206.7060000002));

#21=IFCCARTESIANPOINT((3513051.6502999999,5516221.6858000001));

#22=IFCCARTESIANPOINT((3513037.9591000001,5516199.0738000004));

#23=IFCVERTICALALIGNMENT($,$,$,$,$,$,$,(#24,#26,#28,#30,#32));

#24=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#25);

#25=IFCCARTESIANPOINT((-53.048000000000002,124.48999999999999));

#26=IFCVERTICALALIGNMENTPARABOLA(#27,38.1205);

#27=IFCCARTESIANPOINT((0,126.7));

#28=IFCVERTICALALIGNMENTPARABOLA(#29,39.073499999999996);

#29=IFCCARTESIANPOINT((41.199999999999939,124.48999999999999));

#30=IFCVERTICALALIGNMENTPARABOLA(#31,38.120499999999993);

#31=IFCCARTESIANPOINT((94.248000000000019,126.7));

#32=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#33);

#33=IFCCARTESIANPOINT((135.44800000000001,124.48999999999999));

#34=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#35,#97);

#35=IFCHORIZONTALALIGNMENT($,$,$,$,$,$,$,(#36,#43,#49,#56,#62,#69,#76,#83,#90),-75.932000000000002);

#36=IFCHORIZONTALALIGNMENTLINE(#37,$);

#37=IFCTRIMMEDCURVE(#38,(#41),(#42),,$);

#38=IFCLINE(#39,#40);

#39=IFCCARTESIANPOINT((3512794.1875999998,5516196.0411));

#40=IFCVECTOR($,1);

#41=IFCCARTESIANPOINT((3512794.1875999998,5516196.0411));

#42=IFCCARTESIANPOINT((3512822.4983999999,5516120.2982000001));

#43=IFCHORIZONTALALIGNMENTCLOTHOID(#44);

#44=IFCTRIMMEDCURVE(#45,(#47),(#48),.F.,$);

#45=IFCCLOTHOID(#46,$,1.#INF,30,12.765700000000001,.START\_RADIUS\_INFINITE.);

#46=IFCCARTESIANPOINT((3512825.4851000002,5516112.3074000003));

#47=IFCCARTESIANPOINT((3512822.4983999999,5516120.2982000001));

#48=IFCCARTESIANPOINT((3512827.7928999998,5516108.7104000002));

#49=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#50);

#50=IFCTRIMMEDCURVE(#51,(#54),(#55),.F.,$);

#51=IFCCIRCLE(#52,30.00005917195389);

#52=IFCAXIS2PLACEMENT2D(#53,$);

#53=IFCCARTESIANPOINT((3512853.0425999998,5516124.9108999996));

#54=IFCCARTESIANPOINT((3512827.7928999998,5516108.7104000002));

#55=IFCCARTESIANPOINT((3512862.2401000001,5516096.3556000004));

#56=IFCHORIZONTALALIGNMENTCLOTHOID(#57);

#57=IFCTRIMMEDCURVE(#58,(#60),(#61),.F.,$);

#58=IFCCLOTHOID(#59,$,30,1.#INF,13.333299999999999,.END\_RADIUS\_INFINITE.);

#59=IFCCARTESIANPOINT((3512866.4904999998,5516097.7246000003));

#60=IFCCARTESIANPOINT((3512862.2401000001,5516096.3556000004));

#61=IFCCARTESIANPOINT((3512874.1625999999,5516102.2593));

#62=IFCHORIZONTALALIGNMENTLINE(#63,$);

#63=IFCTRIMMEDCURVE(#64,(#67),(#68),,$);

#64=IFCLINE(#65,#66);

#65=IFCCARTESIANPOINT((3512874.1625999999,5516102.2593));

#66=IFCVECTOR($,1);

#67=IFCCARTESIANPOINT((3512874.1625999999,5516102.2593));

#68=IFCCARTESIANPOINT((3513050.8722999999,5516206.7062999997));

#69=IFCHORIZONTALALIGNMENTLINE(#70,$);

#70=IFCTRIMMEDCURVE(#71,(#74),(#75),,$);

#71=IFCLINE(#72,#73);

#72=IFCCARTESIANPOINT((3513050.8722999999,5516206.7062999997));

#73=IFCVECTOR($,1);

#74=IFCCARTESIANPOINT((3513050.8722999999,5516206.7062999997));

#75=IFCCARTESIANPOINT((3513053.9791999999,5516216.0925000003));

#76=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#77);

#77=IFCTRIMMEDCURVE(#78,(#81),(#82),.T.,$);

#78=IFCCIRCLE(#79,38.000067071740411);

#79=IFCAXIS2PLACEMENT2D(#80,$);

#80=IFCCARTESIANPOINT((3513090.0543,5516204.1513));

#81=IFCCARTESIANPOINT((3513053.9791999999,5516216.0925000003));

#82=IFCCARTESIANPOINT((3513066.3642000002,5516233.8629000001));

#83=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#84);

#84=IFCTRIMMEDCURVE(#85,(#88),(#89),.T.,$);

#85=IFCCIRCLE(#86,99.999941319287529);

#86=IFCAXIS2PLACEMENT2D(#87,$);

#87=IFCCARTESIANPOINT((3513128.7064999999,5516155.6744999997));

#88=IFCCARTESIANPOINT((3513066.3642000002,5516233.8629000001));

#89=IFCCARTESIANPOINT((3513076.6948000002,5516241.0839999998));

#90=IFCHORIZONTALALIGNMENTLINE(#91,$);

#91=IFCTRIMMEDCURVE(#92,(#95),(#96),,$);

#92=IFCLINE(#93,#94);

#93=IFCCARTESIANPOINT((3513076.6948000002,5516241.0839999998));

#94=IFCVECTOR($,1);

#95=IFCCARTESIANPOINT((3513076.6948000002,5516241.0839999998));

#96=IFCCARTESIANPOINT((3513096.8922999999,5516253.3836000003));

#97=IFCVERTICALALIGNMENT($,$,$,$,$,$,$,(#98,#100,#102,#104,#106,#108,#110,#112,#114,#116,#118,#120,#122,#124,#126,#128,#130,#132,#134));

#98=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#99);

#99=IFCCARTESIANPOINT((-75.932000000000002,126.23));

#100=IFCVERTICALALIGNMENTPARABOLA(#101,10);

#101=IFCCARTESIANPOINT((-50.790300000001281,126.10429999999999));

#102=IFCVERTICALALIGNMENTPARABOLA(#103,11.055099999999996);

#103=IFCCARTESIANPOINT((-35.071400000000132,126.1829));

#104=IFCVERTICALALIGNMENTPARABOLA(#105,5.5275999999999996);

#105=IFCCARTESIANPOINT((-1.5686999999995328,125.98));

#106=IFCVERTICALALIGNMENTPARABOLA(#107,32.999200000000002);

#107=IFCCARTESIANPOINT((21.626999999999953,126.096));

#108=IFCVERTICALALIGNMENTPARABOLA(#109,84.002900000000011);

#109=IFCCARTESIANPOINT((150.65600000000001,130.999));

#110=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#111);

#111=IFCCARTESIANPOINT((252.30000000000001,126.32299999999999));

#112=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#113);

#113=IFCCARTESIANPOINT((256.30000000000001,126.139));

#114=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#115);

#115=IFCCARTESIANPOINT((260.65600000000001,125.965));

#116=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#117);

#117=IFCCARTESIANPOINT((265.65600000000001,125.815));

#118=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#119);

#119=IFCCARTESIANPOINT((265.65600000000001,125.815));

#120=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#121);

#121=IFCCARTESIANPOINT((267.65600000000001,125.755));

#122=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#123);

#123=IFCCARTESIANPOINT((275.65600000000001,125.54600000000001));

#124=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#125);

#125=IFCCARTESIANPOINT((283.65600000000001,125.33799999999999));

#126=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#127);

#127=IFCCARTESIANPOINT((285.65600000000001,125.29300000000001));

#128=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#129);

#129=IFCCARTESIANPOINT((290.65600000000001,125.17));

#130=IFCVERTICALALIGNMENTPARABOLA(#131,10.091300000000047);

#131=IFCCARTESIANPOINT((302.87099999999958,125.23099999999999));

#132=IFCVERTICALALIGNMENTPARABOLA(#133,15.252500000000055);

#133=IFCCARTESIANPOINT((332.1279999999993,124.639));

#134=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#135);

#135=IFCCARTESIANPOINT((343.76999999999998,124.581));

#136=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#137,#159);

#137=IFCHORIZONTALALIGNMENT($,$,$,$,$,$,$,(#138,#145,#152),0);

#138=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#139);

#139=IFCTRIMMEDCURVE(#140,(#143),(#144),.T.,$);

#140=IFCCIRCLE(#141,9.0000414031935456);

#141=IFCAXIS2PLACEMENT2D(#142,$);

#142=IFCCARTESIANPOINT((3513050.8722999999,5516206.7060000002));

#143=IFCCARTESIANPOINT((3513058.4393000002,5516201.8334999997));

#144=IFCCARTESIANPOINT((3513043.1244000001,5516202.1267999997));

#145=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#146);

#146=IFCTRIMMEDCURVE(#147,(#150),(#151),.T.,$);

#147=IFCCIRCLE(#148,8.9999459471794303);

#148=IFCAXIS2PLACEMENT2D(#149,$);

#149=IFCCARTESIANPOINT((3513050.8722999999,5516206.7060000002));

#150=IFCCARTESIANPOINT((3513043.1244000001,5516202.1267999997));

#151=IFCCARTESIANPOINT((3513053.7004,5516215.2500999998));

#152=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#153);

#153=IFCTRIMMEDCURVE(#154,(#157),(#158),.T.,$);

#154=IFCCIRCLE(#155,8.9999885785073772);

#155=IFCAXIS2PLACEMENT2D(#156,$);

#156=IFCCARTESIANPOINT((3513050.8722999999,5516206.7060000002));

#157=IFCCARTESIANPOINT((3513053.7004,5516215.2500999998));

#158=IFCCARTESIANPOINT((3513058.4393000002,5516201.8335999995));

#159=IFCVERTICALALIGNMENT($,$,$,$,$,$,$,(#160,#162,#164,#166,#168));

#160=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#161);

#161=IFCCARTESIANPOINT((-15,125.33));

#162=IFCVERTICALALIGNMENTPARABOLA(#163,17);

#163=IFCCARTESIANPOINT((-4.9999999999999893,124.98));

#164=IFCVERTICALALIGNMENTPARABOLA(#165,14.449999999999998);

#165=IFCCARTESIANPOINT((18.564699999999974,126.15819999999999));

#166=IFCVERTICALALIGNMENTPARABOLA(#167,17);

#167=IFCCARTESIANPOINT((51.828200000000002,124.994));

#168=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#169);

#169=IFCCARTESIANPOINT((61.548000000000002,125.48));

#170=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#171,#200);

#171=IFCHORIZONTALALIGNMENT($,$,$,$,$,$,$,(#172,#179,#186,#193),0);

#172=IFCHORIZONTALALIGNMENTLINE(#173,$);

#173=IFCTRIMMEDCURVE(#174,(#177),(#178),,$);

#174=IFCLINE(#175,#176);

#175=IFCCARTESIANPOINT((3512824.3969999999,5516115.6298000002));

#176=IFCVECTOR($,1);

#177=IFCCARTESIANPOINT((3512824.3969999999,5516115.6298000002));

#178=IFCCARTESIANPOINT((3512835.0087000001,5516114.7900999999));

#179=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#180);

#180=IFCTRIMMEDCURVE(#181,(#184),(#185),.F.,$);

#181=IFCCIRCLE(#182,29.999992297324297);

#182=IFCAXIS2PLACEMENT2D(#183,$);

#183=IFCCARTESIANPOINT((3512837.3753,5516144.6966000004));

#184=IFCCARTESIANPOINT((3512835.0087000001,5516114.7900999999));

#185=IFCCARTESIANPOINT((3512867.1367000001,5516140.9205));

#186=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#187);

#187=IFCTRIMMEDCURVE(#188,(#191),(#192),.T.,$);

#188=IFCCIRCLE(#189,12.000033721631718);

#189=IFCAXIS2PLACEMENT2D(#190,$);

#190=IFCCARTESIANPOINT((3512879.0413000002,5516139.4101));

#191=IFCCARTESIANPOINT((3512867.1367000001,5516140.9205));

#192=IFCCARTESIANPOINT((3512889.7474000002,5516144.8304000003));

#193=IFCHORIZONTALALIGNMENTLINE(#194,$);

#194=IFCTRIMMEDCURVE(#195,(#198),(#199),,$);

#195=IFCLINE(#196,#197);

#196=IFCCARTESIANPOINT((3512889.7474000002,5516144.8304000003));

#197=IFCVECTOR($,1);

#198=IFCCARTESIANPOINT((3512889.7474000002,5516144.8304000003));

#199=IFCCARTESIANPOINT((3512891.8927000002,5516140.5928999996));

#200=IFCVERTICALALIGNMENT($,$,$,$,$,$,$,(#201,#203,#205,#207,#209,#211,#213));

#201=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#202);

#202=IFCCARTESIANPOINT((0,126.1478));

#203=IFCVERTICALALIGNMENTPARABOLA(#204,3.3605);

#204=IFCCARTESIANPOINT((6.0552999999999209,125.855));

#205=IFCVERTICALALIGNMENTPARABOLA(#206,4.9018999999999977);

#206=IFCCARTESIANPOINT((19.485999999999969,125.6568));

#207=IFCVERTICALALIGNMENTPARABOLA(#208,5.355000000000004);

#208=IFCCARTESIANPOINT((31.243799999999951,124.90689999999999));

#209=IFCVERTICALALIGNMENTPARABOLA(#210,13.683799999999998);

#210=IFCCARTESIANPOINT((70.710299999999989,120.2765));

#211=IFCVERTICALALIGNMENTPARABOLA(#212,4.0989000000000146);

#212=IFCCARTESIANPOINT((80.752400000000065,120.4725));

#213=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#214);

#214=IFCCARTESIANPOINT((91.599999999999994,120.23950000000001));

#215=IFCREFERENCECURVEALIGNMENT2D($,$,$,$,$,$,$,#216,#259);

#216=IFCHORIZONTALALIGNMENT($,$,$,$,$,$,$,(#217,#224,#231,#238,#245,#252),0);

#217=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#218);

#218=IFCTRIMMEDCURVE(#219,(#222),(#223),.F.,$);

#219=IFCCIRCLE(#220,60.00002727922606);

#220=IFCAXIS2PLACEMENT2D(#221,$);

#221=IFCCARTESIANPOINT((3513097.2546000001,5516242.2472000001));

#222=IFCCARTESIANPOINT((3513041.5564000001,5516219.9376999997));

#223=IFCCARTESIANPOINT((3513046.6206999999,5516210.0569000002));

#224=IFCHORIZONTALALIGNMENTLINE(#225,$);

#225=IFCTRIMMEDCURVE(#226,(#229),(#230),,$);

#226=IFCLINE(#227,#228);

#227=IFCCARTESIANPOINT((3513046.6206999999,5516210.0569000002));

#228=IFCVECTOR($,1);

#229=IFCCARTESIANPOINT((3513046.6206999999,5516210.0569000002));

#230=IFCCARTESIANPOINT((3513060.0438000001,5516188.943));

#231=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#232);

#232=IFCTRIMMEDCURVE(#233,(#236),(#237),.F.,$);

#233=IFCCIRCLE(#234,50.000009410418656);

#234=IFCAXIS2PLACEMENT2D(#235,$);

#235=IFCCARTESIANPOINT((3513102.2387000001,5516215.7681999998));

#236=IFCCARTESIANPOINT((3513060.0438000001,5516188.943));

#237=IFCCARTESIANPOINT((3513071.6261,5516176.2350000003));

#238=IFCHORIZONTALALIGNMENTLINE(#239,$);

#239=IFCTRIMMEDCURVE(#240,(#243),(#244),,$);

#240=IFCLINE(#241,#242);

#241=IFCCARTESIANPOINT((3513071.6261,5516176.2350000003));

#242=IFCVECTOR($,1);

#243=IFCCARTESIANPOINT((3513071.6261,5516176.2350000003));

#244=IFCCARTESIANPOINT((3513098.1771999998,5516155.6752000004));

#245=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#246);

#246=IFCTRIMMEDCURVE(#247,(#250),(#251),.F.,$);

#247=IFCCIRCLE(#248,25.000025904957653);

#248=IFCAXIS2PLACEMENT2D(#249,$);

#249=IFCCARTESIANPOINT((3513113.4835000001,5516175.4418000001));

#250=IFCCARTESIANPOINT((3513098.1771999998,5516155.6752000004));

#251=IFCCARTESIANPOINT((3513110.3635,5516150.6371999998));

#252=IFCHORIZONTALALIGNMENTCIRCULARSEGMENT(#253);

#253=IFCTRIMMEDCURVE(#254,(#257),(#258),.T.,$);

#254=IFCCIRCLE(#255,23.499925494517584);

#255=IFCAXIS2PLACEMENT2D(#256,$);

#256=IFCCARTESIANPOINT((3513107.4306999999,5516127.3210000005));

#257=IFCCARTESIANPOINT((3513110.3635,5516150.6371999998));

#258=IFCCARTESIANPOINT((3513123.2607999998,5516144.6892999997));

#259=IFCVERTICALALIGNMENT($,$,$,$,$,$,$,(#260,#262,#264,#266,#268,#270,#272));

#260=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#261);

#261=IFCCARTESIANPOINT((0,125.8745));

#262=IFCVERTICALALIGNMENTPARABOLA(#263,6.8000000000000007);

#263=IFCCARTESIANPOINT((6.1014999999999997,125.6122));

#264=IFCVERTICALALIGNMENTPARABOLA(#265,12.899999999999999);

#265=IFCCARTESIANPOINT((30.804500000000097,125.38979999999999));

#266=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#267);

#267=IFCCARTESIANPOINT((44.332500000000003,124.68640000000001));

#268=IFCVERTICALALIGNMENTPARABOLA(#269,18.749999999999993);

#269=IFCCARTESIANPOINT((59.542700000000025,124.3822));

#270=IFCVERTICALALIGNMENTPARABOLA(#271,32.621100000000013);

#271=IFCCARTESIANPOINT((98.000000000000043,121.2094));

#272=IFCVERTICALALIGNMENTPOINTVERTICALINTERSCECTION(#273);

#273=IFCCARTESIANPOINT((114.77200000000001,120.92));

ENDSEC;

END-ISO-10303-21;